REMARKS

I. Introduction

In response to the Office Action dated December 4, 2003, please consider the following remarks. Re-examination and re-consideration of the application, as amended, is requested.

II. Office Action Double Patenting Rejection

In paragraph (4), the Office Action provisionally rejects claims 22-36 under the judicially-created doctrine of double patenting as being unpatentable over claims 1-4, 6, 7-10, 12, 13-16, and 18 of U.S. Patent No. 6,332,211.

The Applicant respectfully traverses this rejection, but includes herewith a terminal disclaimer rendering this rejection moot.

III. The Cited References and the Subject Invention

A. The Smith Jr. Reference

U.S. Patent No. 5,754,755, issued May 19, 1998 to Smith discloses a method and system for generating an application-specific test script file. The application-specific test script file contains test instructions for testing an application program. The system receives a test template file that has test instructions that contain placeholders. The placeholders indicate where application-specific placeholder values are to be logically inserted into the test template file. The system receives an ordered list of customizing files that have application-specific placeholder values. The system then searches the customizing files according to the ordered list for a first placeholder value for each placeholder of the test instruction. When such a placeholder value is found, the system replaces the placeholder with the placeholder value in the test instruction and stores the test instruction into the application-specific test script file.

IV. Office Action Prior Art Rejections

In paragraph (5), the Office Action rejected claims 22-24, 26-29, 31-34, and 36 under 35 U.S.C. § 102(a) as anticipated by Smith, Jr., U.S. Patent No. 5,754,755 (Smith).

With Respect to Claims 22, 27, and 32: Claim 22 recites:

A method of generating test code for an automated test procedure applyable to a system comprising a plurality of interconnected elements, the method comprising the steps of:
defining a source file having a plurality of tags, each tag associated with a member of a library of executable code objects defining a set of instructions for performing a portion of the automatic test procedure;
generating a test plan in a conversational language from the source file; and
generating the test code for the automated test procedure from the source file.

According to the Office Action, Smith discloses "defining a source file having a plurality of tags," in elements 112 and 144 of FIG. 1:

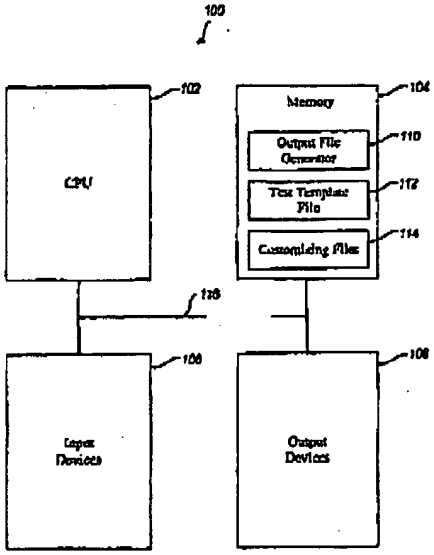

Fig. 1

and in the text cited below:

> For example, the HTML template file might contain placeholders for stock price and volume data. The customizing files stored in the server contain the current price and volume information that would be inserted into the HTML output file (application-specific test script). (col. 7, lines 25-30)

Essentially, the Office Action argues that the "source file" of the Applicant's invention is analogous to the test template file 112 of the Smith reference, and that the "tags" of the Applicant's invention are analogous to the "placeholders".

Referring to the following passage, the Office Action further suggests that each placeholder is "associated with a member of a library of executable code objects defining a set of instructions for performing a portion of the automatic test procedure":

> The ##FOR command indicates that iteration statements follow. The ##NEXT statement indicates the end of the iteration statements that are to be repeated over the range of numerical values from a to b. For each iteration of the iterative command, the indexed placeholder within the iteration statements is replaced by a placeholder with it's asterisk (*) by a value starting with the value of a and sequentially increasing to the value of b. For example, for each iteration of the iterative commands 210, 213, the placeholder takes on the value OBJECT1, OBJECT2, OBJECT3, etc. The output file generator searches the customizing files for a values for the generated placeholders, replaces the generated placeholders with those values, and stores the instructions 211, 212 in the application-specific test script. In addition, "a" and "b" may be placeholders, in which case, their value would be retrieved from a customizing file prior to the evaluation of the #FOR command. (col. 5, lines 49-65).

However, Smith's "placeholders" are not analogous to the "tags" of the Applicant's invention, because the "placeholders" are not *associated with a library of executable code objects (314) defining a set of instructions for performing a portion of the automated test procedure.* Instead, Smith's "placeholders" are precisely that ... placeholders that are *replaced* with application specific *data* (which are essentially arguments) that are obtained from the customizing files (see col. 3, lines 11-13). The placeholders are not associated with a library of executable code objects defining a set of instructions.

An exemplary embodiment of a library of executable code objects is illustrated in the Applicant's specification as element 2108 ("Coupling Facility"), and an exemplary embodiment of library members is illustrated as element group 2102 (e.g. "CF Link Failure Recovery", "CF Structure Failure/Recove", "Issue Operator Rebuild Com", and "Simulate CF Connection Fail") in FIG. 22 of the Applicant's specification, reproduced below:

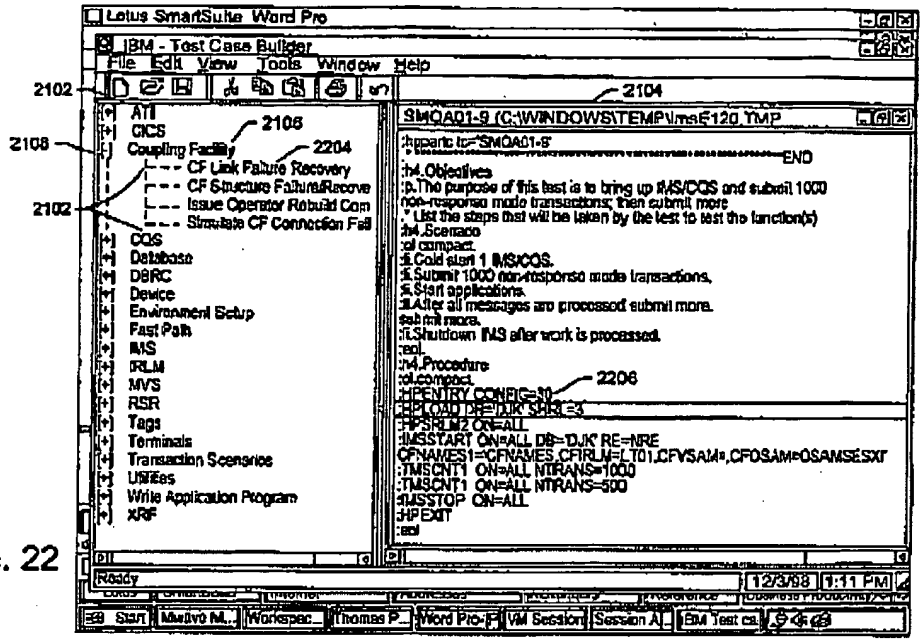

FIG. 22

In contrast, the "placeholders" of Smith are *replaced* with application specific *data* from the customizing files (see col. 3, lines 11-13). Smith provides an exemplary test template file in FIG. 2 below:

```
201   [TEMPLATE]

202   #INCLUDE "WORD.INC" -
203   #INCLUDE "EXCEL.INT"
204   #INCLUDE "COMMON.INC"

205   Sub Auto_Open()

206   Dim szTestDescription As String        'Holds description of test
207   Dim szApplicationName As String        'Holds name of an application program

208   szTestDescription = [{SCRIPT_COMMENT}]

209   szApplicationName = [{APPLICATION}]

210   #FOR [{OBJECT*}] = 1 to 10
211         Insert [{OBJECT*}]
212         Debug.Print [{OBJECT*}]
213   #NEXT

214   #IF [{SPELLCHECKER}] # -# "LINUS" THEN
215         [{SPELLTEST}]
216   #ENDIF
```

Fig. 2

Each of the elements in double brackets [{ }], for example, [{SCRIPT_COMMENT}], [{APPLICATION}], and {[OBJECT}] are placeholders. *Data*, which is obtained from the customizing files, is substituted for the placeholders to generate the test script. For example, the following ("COMMON") customizing file includes *data values* that are substituted for the "SCRIPT_COMMENT" and "SPELLCHECKER" placeholders:

```
3A1   [DATA]      (COMMON.INC)
3A2   SCRIPT_COMMENT = "TESTSCRIPT"
3A3   SPELLCHECKER = "FALSE"
```

Fig. 3A

and the following ("DATA") customizing file includes *data values* that are substituted for the "APPLICATION" and "OBJECT*" placeholders.

```
3B1   [DATA]      (EXCEL.INC)

3B2   APPLICATION = "EXCEL"
3B3   OBJECT1 = "MICROSOFT WORD DOCUMENT"
3B4   OBJECT2 = "MICROSOFT WORD PICTURE"
3B5   OBJECT3 = "MICROSOFT POWERPOINT SLIDE"
```

Fig. 3B

The result (and the application of the *data values* shown in FIG. 3C, which are not reproduced here in the interest of brevity) is the code shown below:

```
Sub Auto_Open()

Dim szTestDescription As String          'Holds description of test
Dim szApplicationName As String          'Holds name of an application program

szTestDescription = "TESTSCRIPT"

szApplicationName = "WORD"
Insert "MICROSOFT EXCEL WORKSHEET"
Insert "MICROSOFT EXCEL CHART"
Insert "MICROSOFT POWERPOINT SLIDE"
Insert "MICROSOFT POWERPOINT PRESENTATION"
spell ("supersede")
spell ("qoph")
spell ("pteridology")
spell ("parmoffa")
spell ("lassitude")
```

Fig. 4

Unlike the Smith reference which teaches the substitution of *placeholders* with *data values* from a customizing file, the Applicant's invention teaches the use of tags which are associated with a library of executable code objects defining a set of *instructions* for performing a portion of the automated test procedure.

The foregoing differences are a product of the different purposes to which they are directed. The Smith reference offers a testing mechanism that permits use of a single test script for different application programs, but allows for differences between the programs (see col. 2, lines 19-22). Thus, it provides a template that can be used with multiple application programs and includes placeholders that are replaced with different input values (each for a different application program). The Applicant's invention is not directed to creating a test program that is applicable to different applications, but rather, a system that guides generation of test cases, and thus, provides tags associated with a library of executable code objects that can be used to debug the code.

For all of the foregoing reasons, the Applicant respectfully traverses the rejection of claim 22.

Claim 27 and 32 recite features analogous to those of claim 22, and are patentable for the same reasons.

With Respect to Claims 24, 29, and 34: Claim 24 recites:

> *The method of claim 23, wherein the test plan comprises a test index identifying the system elements tested by the test code, the test index generated by performing the step of scanning the interpreted tags to identify the system elements tested by the test code.*

According to the Office Action, the Smith reference discloses the generation of a text index as follows:

> When generating an application-specific test script, the output file generator scans through the customizing files in order searching for a value for each placeholder. (col. 3, lines 45-48)

However, this does not disclose the generation of a "test index identifying system elements tested by the test code" nor does it disclose performing this task by "scanning the *interpreted* tags to identify the system elements tested by the test code," as recited in claim 22.

Claims 29 and 34 recites features analogous to those of claim 24, and are patentable for the same reasons.

With Respect to Claim 26, 31, and 36: Claim 26 recites:

> *The method of claim 22, wherein the step of generating test code for the automated test procedure comprises the step of translating the executable code objects associated with the tag in the source file.*

As described above, Smith does not disclose tags associated with executable code objects, and therefore cannot disclose translating the executable code objects associated with tags. Accordingly, the Applicant traverses the rejection of claim 26. Claims 31 and 36 recite features analogous to those of claim 26 and are patentable for the same reasons

In paragraph (6), the Office Action rejected claims 25, 30, and 35 under 35 U.S.C. §103(a) as being unpatentable over Smith. Applicant respectfully traverses these rejections.

Claim 25 recites:

> *The method of claim 23, wherein the step of generating a test plan further comprises the steps of:*
> *identifying an uninterpretable tag in the test plan; and*
> *appending the test plan with an error message identifying the uninterpretable tag.*

The Office Action argues that the foregoing is "syntax" error checking, and that such checking is well known in the art. However, recalling that in rejecting claim 22, the Office Action analogized "tags" to "placeholders", which, as described above, is a placeholder for *data*, and *data* is not "uninterpretable."

The foregoing difficulty, of course, is a product of the erroneous premise that the Applicant's "tags" are analogous to Smith's "placeholders." Nonetheless, since one of ordinary skill in the art would not perform a syntax error check on a placeholder for "data," the Applicant traverses this rejection.

Claims 30 and 35 recite limitations analogous to those of claim 25, and are patentable for the same reasons.

V. Dependent Claims

Dependent claims 23-26, 28-31, and 33-36 incorporate the limitations of their related independent claims, and are therefore patentable on this basis. In addition, these claims recite novel elements even more remote from the cited references. Accordingly, the Applicant respectfully requests that these claims be allowed as well.

VI. Conclusion

In view of the above, it is submitted that this application is now in good order for allowance and such allowance is respectfully solicited. Should the Examiner believe minor matters still remain that can be resolved in a telephone interview, the Examiner is urged to call Applicant's undersigned attorney.

Respectfully submitted,

GATES & COOPER LLP
Attorneys for Applicant(s)

Howard Hughes Center
6701 Center Drive West, Suite 1050
Los Angeles, California 90045
(310) 641-8797

Date: March 3, 2004

By: _____
Name: Victor G. Cooper
Reg. No.: 39,641

VGC/io